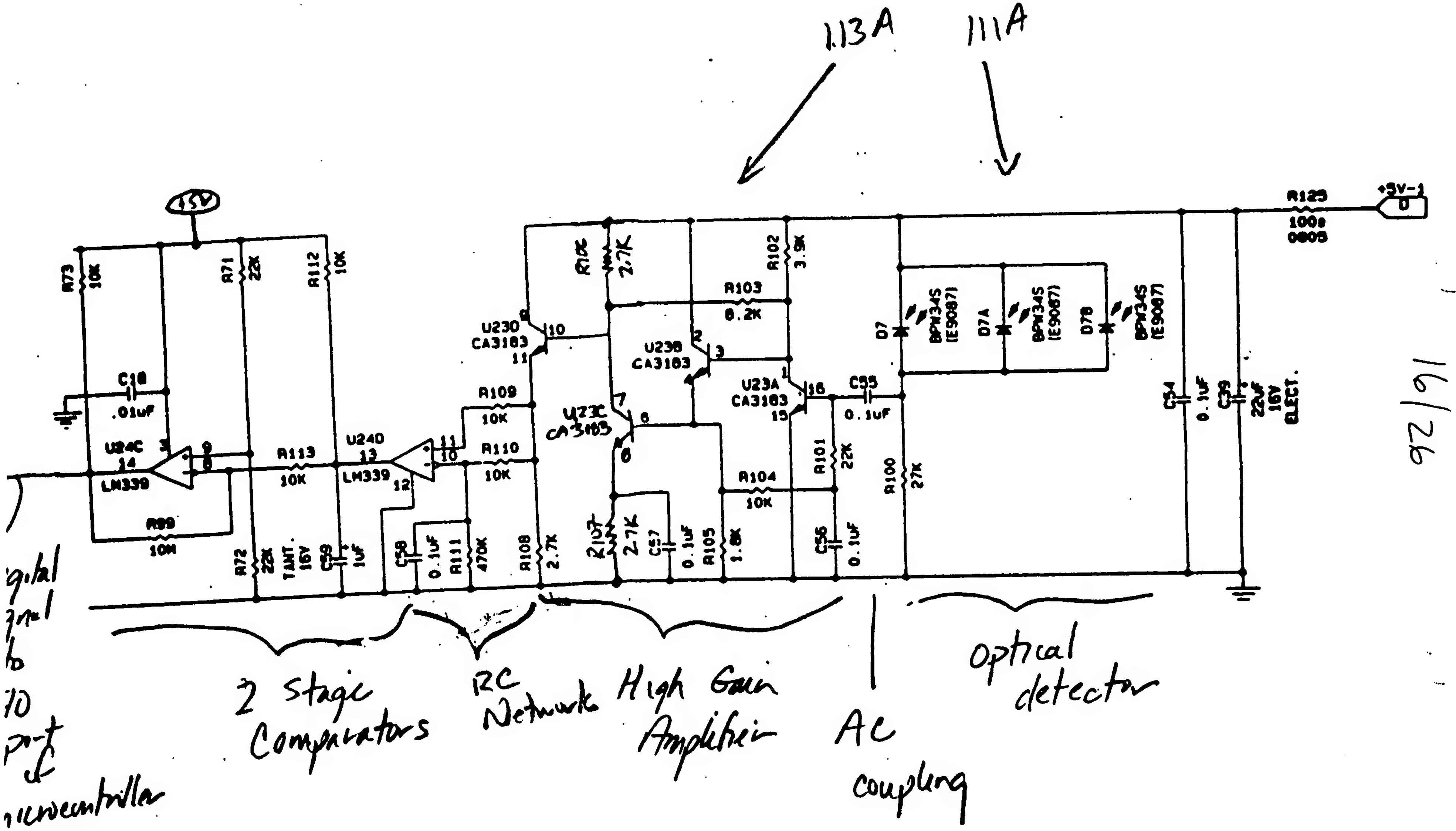


~~FIG. 7(A)~~
FIGS. 7(A)(i) and 7(A)(ii)

```
init:
    Direction = UP;
main_loop:
    if mode_switching = ON  then:
        if Direction = UP   then
heat_loop:
                heat laser light source (set PW = 100%)
                    if mode_switching = OFF  then
                        calculate new_PW to maintain temp
                        set PW to new_PW
                        jump to main_loop
                    else
                        if top_of_range_reached then
                            Direction = DOWN;
                            jump to cool_loop;
                        else
                            jump to heat_loop;
                        end if;
                    end if;
        else  /***** Direction  = DOWN *****/
cool_loop:
                cool laser light source (set PW = 0%)
                    if mode_switching = OFF then
                        calculate new_PW to maintain temperature;
                        set PW to new_PW
                        jump to main_loop
                    else
                        if bottom_of_range_reached then
                            Direction = UP;
                            jump to heat_loop;
                        else
```

FIG ~~8(A)(i)~~ 8(A)

```
                                    jump to cool_loop;
                                end if;
                            endif;
        else
            use PW  to maintain temperature
            jump to main_loop
        endif;
    end
```

FIG ~~8(A)(ii)~~ 8(A)

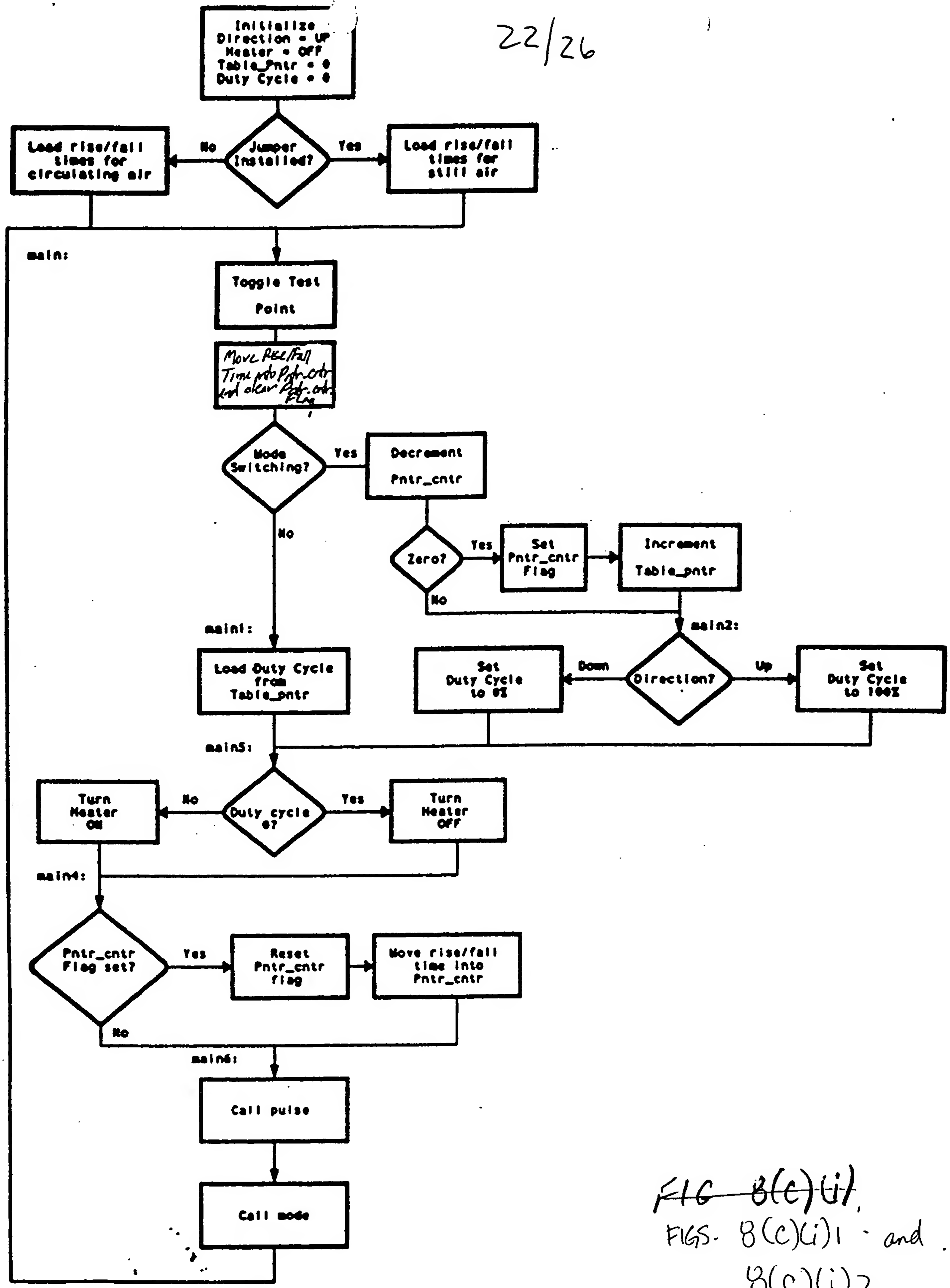
```
main_loop:
    if mode_switching = ON  then begin:

        if heater_power = low  (PW <= 50%)  then
heat_loop:
                heat laser light source (set PW = 100%)
                    if mode_switching = OFF  then
                        calculate new_PW  to maintain temp
                        set PW to new_PW
                        jump to main_loop
                    else
                        if top_of_range_reached then
                            jump to cool_loop;
                        else
                            jump to heat_loop;
                        end if;
                    end if;
        else  /***** heater_power = high  (PW >50%)  ****/
cool_loop:
                cool laser light source (set PW = 0%)
                    if mode_switching = OFF then
                        calculate new_PW to maintain temperature;
                        set PW to new_PW
                        jump to main_loop
                    else
                        if bottom_of_range_reached then
                            jump to heat_loop;
                        else
                            jump to cool_loop;
                        end if;
                    endif;
        endif;
```

FIG. ~~8(B)(i)~~ 8(B)

```
    else
        use PW  to maintain temperature
        jump to main_loop
    endif;
end
```

FIG. ~~8(B)(ii)~~ 8(B)



~~FIG 8(c)(i)~~,
FIGS. 8(c)(i)1 and
8(c)(i)2

```
;**********************************************************************

        list      p=12c509            ; list directive to define processor
        #include <p12c509.inc>        ; processor specific variable definitions

        __CONFIG   _CP_OFF & _WDT_OFF & _MCLRE_OFF & _IntRC_OSC

; '__CONFIG' directive is used to embed configuration word within .asm file.
; The lables following the directive are located in the respective .inc file.
; See respective data sheet for additional information on configuration word.
        labels

;***** VARIABLE DEFINITIONS
;Labels for variables
threshold       EQU     0x25    ;set threshold level for mode switching
modeswitch      EQU     0x03    ;Input signal location
heater          EQU     0x00    ;Output signal location
TP              EQU     0x02    ;Test Point location
rise1           EQU     D'120'  ;first rise time (120*2 seconds) jumper IN
rise2           EQU     D'45'   ;second rise time (45*2 seconds) jumper OUT
fall1           EQU     D'120'  ;first fall time (120*2 seconds) jumper IN
fall2           EQU     D'45'   ;second fall time (45*2 seconds) jumper OUT

;Labels for memory locations
temp            EQU     0x07    ;example variable definition
duty_cycle      EQU     0x08    ;Pulse width modulation
modeswitch_255  EQU     0x09    ;counter to keep track of mode switching
timer0          EQU     0x0a    ;keep track of timer changes
rise            EQU     0x0b
fall            EQU     0x0c
table_pntr      EQU     0x0d
flags           EQU     0x0e
pntr_cntr       EQU     0x0f


;**********************************************************************
                ORG     0x3FF             ; processor reset vector
; Internal RC calibration value is placed at location 0x1FF by Microchip
; as a movlw kk, where the kk is a literal value.

        ORG     0x000            ; coding begins here
        movwf   OSCCAL           ; update register with factory cal value

; remaining code goes here


;***************INITIALIZE
```

FIG. 8(D)(i)

```
        MOVLW   0x3a            ;set up I/O
        TRIS    6

        CLRF    duty_cycle      ;set initial duty cycle to 0
        BCF     GPIO,heater     ;turn  ff heater
        BSF     GPIO,heater     ;turn off heater drive transistor

        MOVLW   rise1           ;Initialize rise and fall times to
        BTFSC   GPIO,5          ;setting setting, predetermined constants
        MOVLW   rise2
        MOVWF   rise
        MOVWF   patr_cntr       ;initialize with rise time

        MOVLW   fall1
        BTFSC   GPIO,5
        MOVLW   fall2
        MOVWF   fall

        CLRF    flags
        CLRF    table_patr


;===============MAIN LOOP

main:
        BSF     GPIO,TP         ;Toggle test point
        BCF     GPIO,TP

        BCF     flags,1         ;clear patr_cntr flag

        BTFSS   flags,0         ;test mode switch flag
        GOTO    main1           ;jump if not set

        DECFSZ  patr_cntr,1     ;if not 0, skip
        GOTO    main2
        BSF     flags,1         ;set patr_cntr flag
        INCF    table_patr      ;advance through table

main2:
        MOVLW   0xff            ;load 'up' direction
        MOVWF   duty_cycle      ;set for up direction
        BTFSC   table_patr,5    ;if in 'up' direction, skip
        CLRF    duty_cycle
        GOTO    main5
main1:
        MOVF    table_patr,0    ;load table pointer in working register
        ANDLW   0x3f            ;strip off higher order bits
        CALL    table           ;fetch duty cycle from lookup table
        MOVWF   duty_cycle      ;load in duty cycle
main5:
        MOVF    duty_cycle,0    ;read in duty cycle
        BTFSS   STATUS,Z        ;if nonzero goto main3
        GOTO    main3
;       BCF     GPIO,0          ;if zero, turn OFF output
        BSF     GPIO,heater     ;if zero, turn OFF heater drive transistor
        GOTO    main4
main3:
;       BSF     GPIO,0          ;turn ON output
        BCF     GPIO,heater     ;turn ON heater drive transistor
main4:
        BTFSS   flags,1         ;if flag is set, reset patr_cntr
        GOTO    main6
        MOVF    rise,0          ;reset patr_cntr
        BTFSC   table_patr,5
        MOVF    fall,0
        MOVWF   patr_cntr
main6:
        CALL    pulse           ;pulse width modulation subroutine
        CALL    mode            ;update modeswitching, set mode bit
        GOTO    main            ;go back to main routine
```

~~FIG. 8(D)(ii)~~

FIGS. 8(D)(ii)(a) and 8(D)(ii)(b)

```
;****************SUBROUTINES

mode:                                   ;include mode switching detect etc.
        BCF     flags,0                 ;clear mode switching flag
        MOVLW   threshold               ;put threshold value in accumulator
        SUBWF   modeswitch_255,0 ;compare
        BTFSC   STATUS,C                ;if modeswitch_255>threshold
        BSF     flags,0                 ;set flag0
        RETLW   0                       ;set flag


;Subroutine to generate pulse width modulation, monitor mode switching
;Prescaler set to 256 Therefore each pass is 256 usec. 256 passes produces
;65 ms basic period for mode switching.

pulse:
        CLRF    modeswitch_255  ;Initialize mode switching register
pulse1:
        INCF    TMR0,0          ;wait until TMR0 increments past 0xff
        BTFSC   STATUS,Z
        GOTO    pulse1
pulse1a:
        MOVF    TMR0,0          ;load timer into W
        MOVWF   timer0          ;put in timer0 monitor

        MOVF    timer0,0        ;move timer0 monitor into W
        SUBWF   duty_cycle,0    ;compare duty cycle with timer0
        BTFSS   STATUS,C        ;if borrow occurs, then
        BCF     GPIO,heater     ;clear output
        BSF     GPIO,heater     ;turn OFF heater drive transistor

        INCFSZ  timer0,0        ;if timer = 255, exit from loop
        GOTO    pulse2
        RETLW   0

pulse2:
        BTFSC   GPIO,modeswitch ;If GP3 is high, then
        INCF    modeswitch_255,1 ;increment modeswitch
pulse2a:
        MOVF    timer0,0        ;put timer0 in W
        XORWF   TMR0,0
        BTFSC   STATUS,Z
        GOTO    pulse2a
        GOTO    pulse1a


;****************TABLES
        radix   dec

table:
        addwf PCL
        dt 0,24,46,66,84,100,115,128,140.151.161.170,178,186,192.198
        dt 204,208,213,217,220,224.227.229.232.234,236,238,239,241,242.255
        dt 255,231,209,189,171.155.140,127,115,104,94,85,77,69,63,57,51,47
        dt 42,38,35,31,28,26,23.21.19,17,16,14,13,0


                end                             ; directive 'end of program'
```

~~FIG 8(D)(iii)~~
FIGS. 8(D)(iii)(a) and
8(D)(iii)(b)